The Victorian 6502 User Group Newsletter

KAOS

For People Who Have Got Smart

HARDWARE: David Anear SOFTWARE: Jeff Rae PASCAL: Tony Durrant
NEWSLETTER: Ian Eyles EDUCATION: Mike Pekin
AMATEUR RADIO: Rod Drysdale VK3BYU 337 0425

ADDRESS ALL CORRESPONDENCE TO 10 FORBES ST., ESSENDON, VIC. 3040

Registered for posting as a publication Category B

Vol.2 No.2 November 1981

In Rod Drysdale's report of the last meeting he mentions OTC's link-up with the American SOURCE. Last week we heard that an Australian SOURCE will be set-up early in 1982. The joining fee (once only) will be $100.00,($60.00 until February 1982) and the charge per hour $10.00, 8am to 6pm and $4.50, 6pm to 8am. Details still have to be confirmed, but we believe that as a User group, our members can join for $35-$40 depending on how many members enroll and we believe there will also be a discount on the hourly rate, also Modems may be available at a discount, but this will not be decided till after the newsletter is printed. We will have further details early in December, so contact us if you are interested.

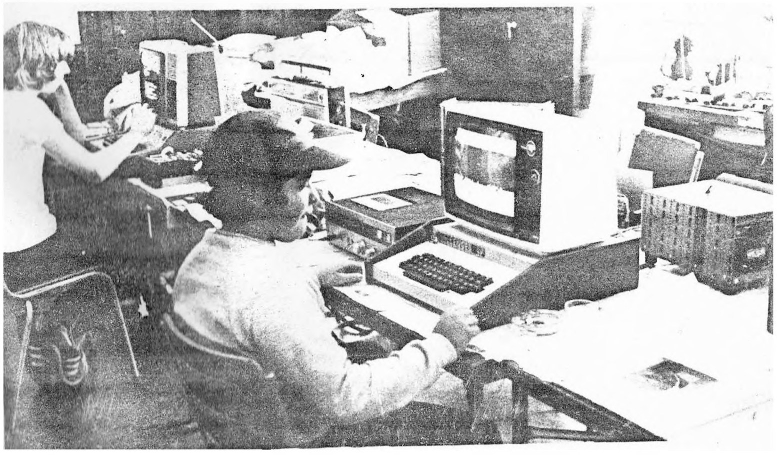
At the last meeting some SYM owners got together and decided to form their own special interest group. More details inside.

We still have some teletypes available for $60.00 each, contact Jeff Rae.

I must be looking over-worked as offers of help are coming from everywhere. Jannene is busy indexing our magazine library and as from mid December Ron Cork will be looking after our program library. We are still looking for programs, parts of programs or ideas for our library. So if you've started writing a program and it still needs some work, send it along, we will find someone to finish it for you.

Memberships are due in January, would you please answer all the questions on the renewal form so that we can up-date our records.

The next meeting will be held at 2 pm on Sunday 29th November at the Essendon Primary School. Would the usual early arrivers please note that the children will be there at 10.30 am.



THE MORNING SESSION AT THE OCTOBER MEETING. Photos by Jeff Kerry.

R G B COLOR

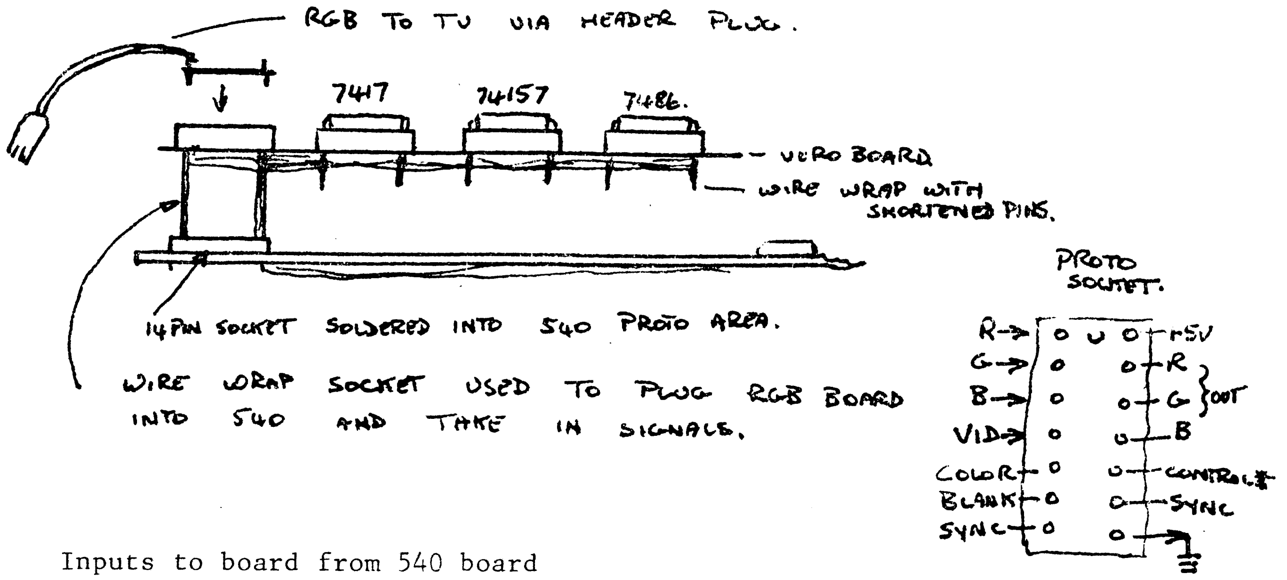
Here is a simple way of getting R G B drive from a 540 board to a color TV.

<u>CAUTION</u>: As lethal voltages are present, do not attempt to modify a color TV unless you know exactly what to do.

When colour is selected by the latch at $DE00 it switches the 74157 MUX supplying RGB exclusively ORed with the video. This gives the same sequence of colours as the OSI, however yellow replaces olive green in the list.

During B/W selection the MUX feeds the video directly to the guns, thereby giving normal B/W .

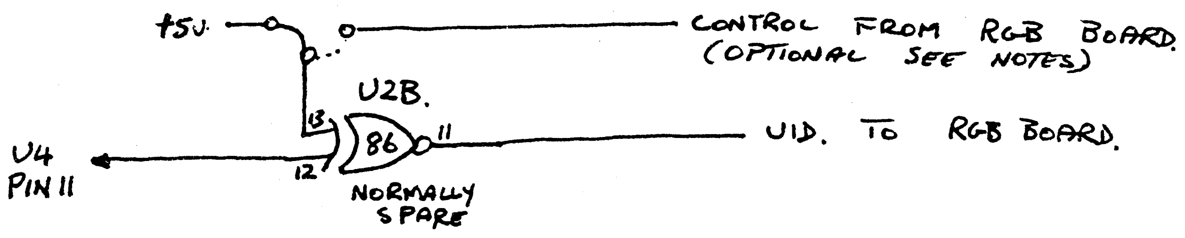
The proto-type was made on a piece of VERO board, which was plugged into the 540 proto socket via a wire wrap socket ie.



RGB TO TV VIA HEADER PLUG.
7417
74157
7486.
VERO BOARD
WIRE WRAP WITH SHORTENED PINS.
14 PIN SOCKET SOLDERED INTO 540 PROTO AREA.
WIRE WRAP SOCKET USED TO PLUG RGB BOARD INTO 540 AND TAKE IN SIGNALS.
PROTO SOCKET.
R
G
B
VID
COLOR
BLANK
SYNC
+5V
R
G
B
OUT
CONTROL
SYNC

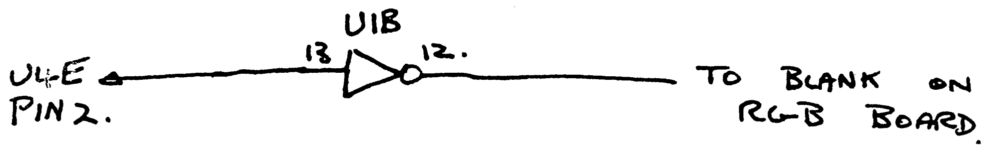
Inputs to board from 540 board

Red	from	U1C	pin	1
Green	"	U1C	"	4
Blue	"	U1c	"	10
Video	use circuit below			



Color	from	U2A pin 13
Sync	from	U4d pin 6

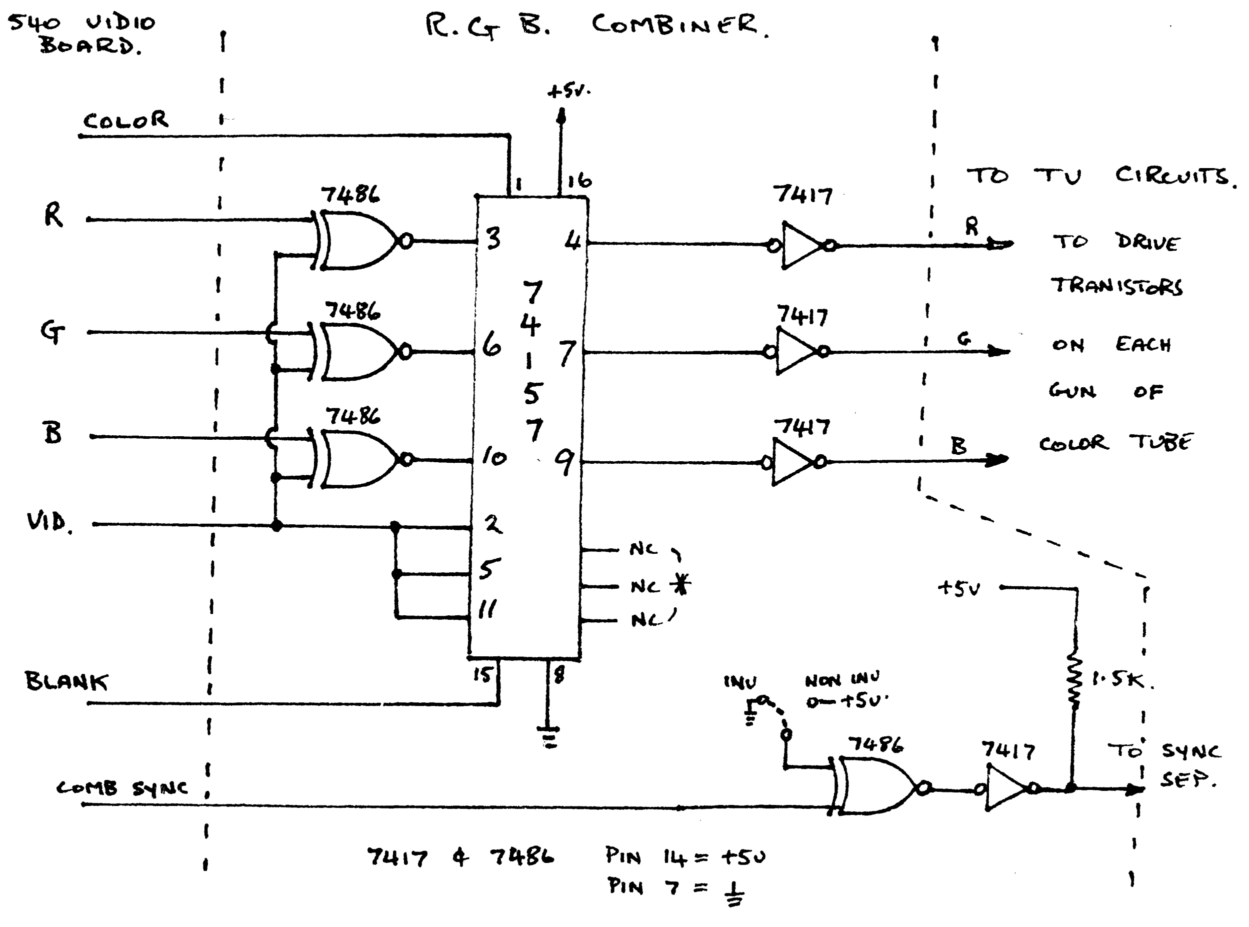
BLANK This presented a problem as it needs to be inverted for our use. Use this circuit:-

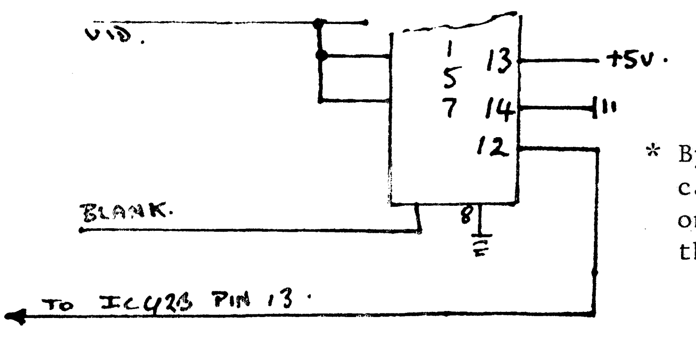


It will be seen that U1B pins 13 and 12 are already used to develop the NTSC delay loop. So cut the tracks going to pins 13 and 12 on foil side of board. Also cut track going to U1B pin 13 from pin 10 on component side of board

OUTPUTS from RGB board

R	Red drive to base of transistor driving tube (must not exceed 30V DC otherwise optical isolation required.)
G	Green drive to gun transistor
B	Blue drive to gun transistor
CONTROL	OPTIONAL-Used to reverse video by switching U2B pin 13 during colour. See circuit diagram for further details.
SYNC	To SYNC separator circuit in T.V.





* By using this spare MUX gate, video can be inverted during color output only. The color sequence is no longer the same as OSI, but it looks better.

SOME USEFUL HINTS

If you are storing programs on cassette then you will notice that when you go to load programs, on completion of the load the tape just happily keeps turning giving rubbish on the screen. If you are using the FILENAME routine by George or the one program per cassette method then at the end of the load you can get the computer back to the keyboard automatically.

When a program has just finished being SAVED type POKE 515,0 and return. Give a short pause and stop the cassette. The effect this has is that when you next load the program the POKE statement is read in and control is returned to the keyboard.

Often it is desired to have the computer just loop waiting for a key to be pressed. One of the simplest ways I have found is to use the following statement. (600 board only)

```
100  IF PEEK(57088) <> 126 THEN 100
```

This statement will cause the program to loop until the REPEAT key is pressed. When the REPEAT key is pressed the processing will continue on the next statment.

Similar statments can be constructed for the SHIFT, CONTROL and ESCAPE keys by substituting the proper value in place of 126. The same can also be done with the 540 board but again with the use of the appropriate values.

Kelvin Eldridge

++

KAOS LIBRARY

KAOS now has a magazine library - albeit small.

We are keeping copies of computing magazines such as Practical Computing, Personal Computer World, Personal Computing etc. At present we have a limited amount of magazines, so to any KAOS member who may be able to part from his magazines once they have been read, don't throw them away, donate them to your library.

To enable us to build a larger collection we are imposing a small borrowing fee of ten cents. To borrow a magazine you can either ring the library and arrange to have the required magazine brought to the next meeting, or arrange at meetings to do the same for the next meeting. Magazines may also be picked up from the library direct.

At meetings contents pages of magazines will be available for your perusal, and some magazines will be available at all meetings for immediate borrowing if you wish.

Future KOAS magazines will have articles about the library, I hope to bring you some library news each month.

For any queries reqarding the KAOS library ring Jannene on .

SOFTWARE FOR THE SIMPLE EPROM PROGRAMMER

by Tony Van Bergen

There are many possible ways to write the software for an Eprom Programmer, ranging from a simple burn only program to sophisticated ones that allow you to program as little as one byte from any where in memory. Its really up to you! I took a middle of the road approach and settled for a fixed 2K area of $2000-27FF (I have 16K RAM) and a few user functions. The programmer sits in ROM beginning at $E400 and uses approx 660 bytes. Entry is from the Monitor, or from the Extended Monitor by typing 'U'.

The functions that I think should be in a programmer program are:

PROG: A four key entry to avoid accidentally writing over another Eprom program. In my program, 2048 bytes take just under 2 mins and then goes straight into:

'C': Compares the contents of the EPROM with memory, counts the differences and prints the total as a 4 digit Hex number. (Sometimes a second burn is needed).

'T': TEST: Runs thru the EPROM testing for $FF. If all 2K locations are $FF, then prints out 'EPROM CLEAR', otherwise prints 'EPROM NOT CLEAR'.

'RE': READ: A 2 key entry because the contents of the EPROM are moved to memory, overwriting whatever was there before.

'D': DUMP: Not essential, but sometimes handy. Prints all errors in the format:-

3 byte address, memory byte, EPROM byte. At the end goes to the count errors routine.

'X': Exits to where ever you wish, for me, the Extended Monitor

SUBROUTINES

Any subroutine that has more than about 10 bytes and is used more than twice is worth considering as a subroutine. The subroutines that I used, briefly, are:

INITIALIZE Used at the start of each subprogram:Sets up the PIA Device (which can be an AY-3-8910 PSG, 6520, 6522, 6530 or 6532), resets the CMOS counter to zero, ensures the 'PROG/READ' switch is in the READ position, sets a zero page DATA POINTER to point to the start of the selected RAM area, and sets a pair of zero page locations to $0800 (2048 decimal) to act as a LOOP COUNTER.

ADJUST Sends a pulse to the CMOS counter to increment its count, increments the RAM DATA POINTER, decrements the LOOP COUNTER then tests for zero in the LOOP COUNTER. The X-register is set to $00 if the counter is not zero, and to $FF if it is. The main programs test for $FF to see if all 2K locations are complete (In the case of PROG and D the program branches to the Count subprogram, in others it goes to the restart).

DELAY A software loop that uses X and Y-registers. to give approx 55mS delay.

MESSAGE Load the X-register with the lo address byte of the desired message which then prints from $E600,X till a null is met (Limitation on this routine: message in $E600-E6FF, end in a null, less than 255 bytes).

BYTE PRINT A routine that prints the contents of the accumulator onto screen as two Hex bytes.

EXIT Prints a shortened input message then jumps back to the restart entry.

GETKEY Calls two subroutines in the standard Monitor to fetch a key from the keyboard and echo it to the screen.

MAIN PROGRAM

On initial entry, clears the screen, then INITIALIZEs to set PROG/READ to READ. Then (and on later reentry) calls GETKEY and branches to the appropriate subprogram

As an example of the flow of logic, I'll describe only the PROG function. Copies of the dissassembled and commented program will be available to any one interested soon.

PROG Calls INITIALIZE, prints'SWITCH TO 'PROG', tests the switch till it is 'PROG', then calls DELAY for switch debounce and circuit settling times. Port 'A' is then set to 'output', a byte is fetched from memory and latched to Port 'A'. A high level is sent thru Port'B' to the EPROM (pin 18)and DELAY called. The program pulse is removed and ADJUST is called. The program then branches back to get another byte from memory, or else jumps to the Count routine.

I hope this gives some members a start to machine code programming. I am no great programmer, so I thank all the people who did the original work in many different programs whose ideas I managed to adapt to my use. Its not easy, but you get a bigger sense of achievement when your program does what you want! Good luck, you'll probably need lots of it!!!

****

INSTANT PROGRAMS:

One of the ideas I have been throwing around for some time is that of a cartridge system for the Superboard. Not having time to pursue the idea I have decided to throw the idea into the laps of you fellow KAOS members. It is time then to put on your thinking caps.

What would you want from a cartridge system? Do you want a mini Tasker Buss where there is only one slot which can fit an EPROM cartridge with resident program or possibly some other arrangement? If you have any ideas what-so-ever then bring them along to the next meeting and we'll have an all out discussion on the topic.

Think about the idea of a cartridge system, it may even be a total waste of time or it may be that time saver to end all time savers. If you can't make the meeting then dump your ideas or comments onto paper and send them to KAOS.

Kelvin Eldridge.

Most of you will have a mysterious piece of software called an Assembler in your collection. A discovery was made about two to three weeks ago which showed us that the OSI Assembler you might posess is a more powerful piece of software than we originally thought.

Besides myself, many people have attempted to unscramble the operations of this marvelous item. When you disassemble (the process of converting pure binary code to original assembler mnemonics) the binary, code.(ie. The form in which the assembler is sold.) You notice sections of meaningful 6502 code, but approximately 70% of the assembler is what appears to be jibberish. Usually, when you see such code, you automatically think that it must be tables which the assembler uses in its' operation. It's most likely the assembler does have tables - but not 70% of its' code. Another difficulty which most of us encountered was when we tried to work out what the 6502 code was doing, we easily got lost in the repeated calls to subroutines. After about four or five subroutines you begin to lose track of where you are up to.

At last, by luck, we struck on a clue which helped in the revealing of the assembler mystery. To understand how the assembler works, you must also understand the concept of an interpreter. Yes, your assembler is an interpreter just like your Basic in Rom. It appears that the actual assembler was originally written for another 16 Bit CPU machine or a 16 Bit pseudo machine. (See later article on 16 Bit pseudo machine.) What OSI have done is add approximately 1K of 6502 code which interprets this 16 Bit CPU's op-codes.

What we have found in this interpreter is a much more powerful machine code instruction set. To give you an idea, there are instructions to do 16 Bit adds and subtracts, and one instruction which does a character compare with a register and if the characters are the same, does a 8 Bit relative jump to a routine. eg. CMP'A',DOASSEM and many more too numerous to show. Also, there must be many more still to be discovered.

We now know how you can use these powerful instructions within your own machine code programs. I have coded a substantial part of the assembler, but because of other commitments I haven't touched it for about a week. So we are asking for volunteers who want to pursue this discovery further. If you are interested in machine code, this discovery makes very interesting reading. I can supply you with all the listings, with comments, and where we are up to, and where to go from here.

George Nikolaidis

****

ATARI NEWS:

KAOS members (who are fully paid members) can now buy Atari products from Kelvin Eldridge (Trading as Personal Computers) with a club discount of five per cent off the retail price. Initially Kelvin's stock will be limited and most items will have to be ordered.

****

OOOOPS!!

Error in the Hex Dump program, KAOS mag. Vol.1 #6 page 10.
First program:
Line 120 >17 should be >16

Second program:
Line 70 >17 should be >16

John Whitehead

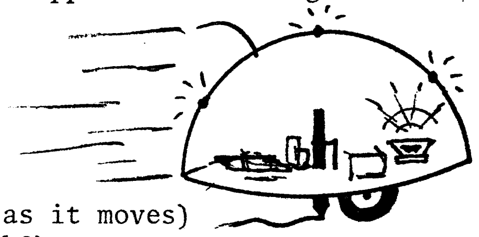
ROBOTS AND EDUCATION
by Jeff Kerry

Quite apart from the fun of having your computer control a mobile machine, a turtle robot can be a great way to teach kids (and others!) about computers and programming.

Most schools in Victoria are already running 'Computer Awareness' courses. But programming involves many abstract ideas, which most people find initially difficult. So a computer with a "friendly", real, moving, physical object as its output device can be an excellent way to discover what a computer does as a result of your instructions.

The Hardware involved is fairly easy. My homemade prototype, run by a Superboard, has been on display at recent Club meetings, and has been tried enthusiastically by some of the primary school kids at the Sunday morning sessions.

The Software is more of a problem. It's a limited version of the "LOGO" language originally devised at Massachusetts Institute of Technology for use with Turtles in schools. If anyone reading this has any information about LOGO or similar languages, I would very much appreciate hearing from them - please contact the Club.

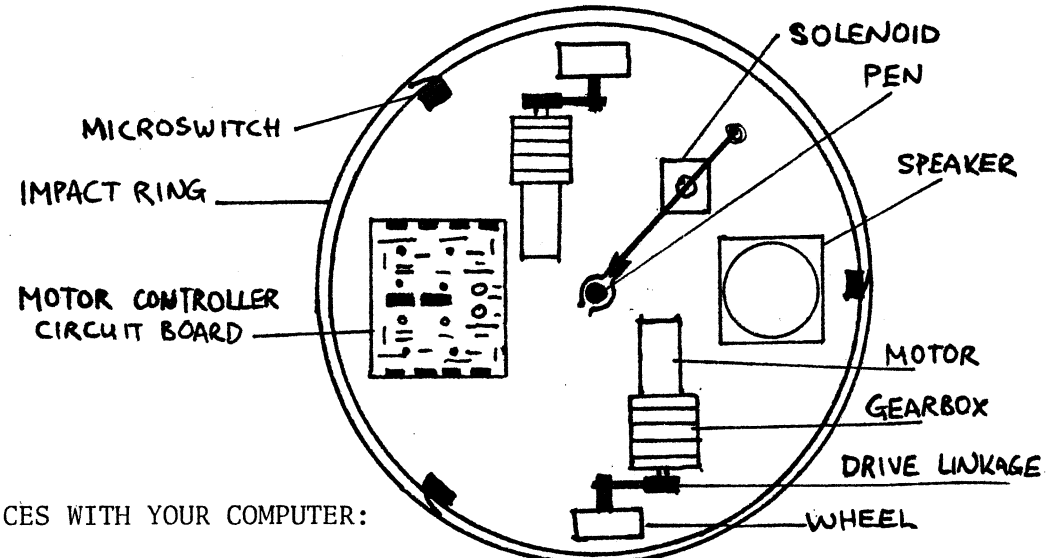


BUILDING A TURTLE:
Capabilities:

- Move forwards or backwards
- Turn (rotate) left or right
- Raise or lower a pen (to write as it moves)
- Flash lights, emit sounds (speak?)
- Detect collision and respond (stop or back off)

All these can be controlled through a PIA port and buffer.

Fig. 1



CONTROLLING DEVICES WITH YOUR COMPUTER:

See Fig.2

The PIA outputs need drive no more than the LEDs in a set of opto-isolators (for safety.) The opto-isolator transistors switch on robust power transistors, which have fuses in their outputs (yes, I learned the hard way!)

This configuration can be used to enable your computer to switch on or off ANY low-voltage devices, eg. motors, lamps, speakers (if you switch them fast enough.) If you use a suitable relay and VERY CAREFUL wiring, you could control 240vAC devices too.

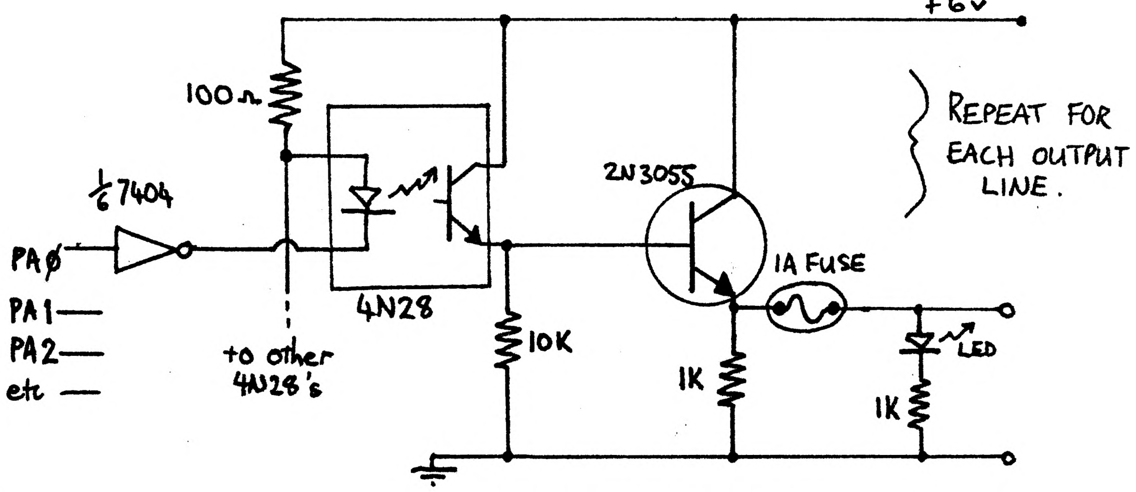
To switch on any device, simply POKE the appropriate number into the PIA location (once you have initialized the PIA.) To turn the device off, POKE a zero.This can even be done with a simple BASIC program

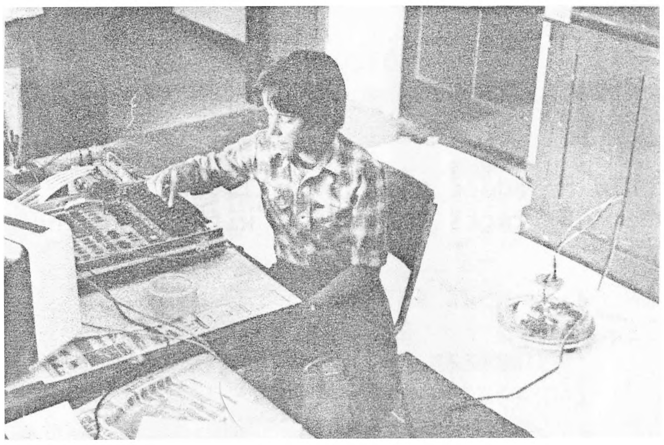
CONTROLLING DC MOTORS: See Fig.3

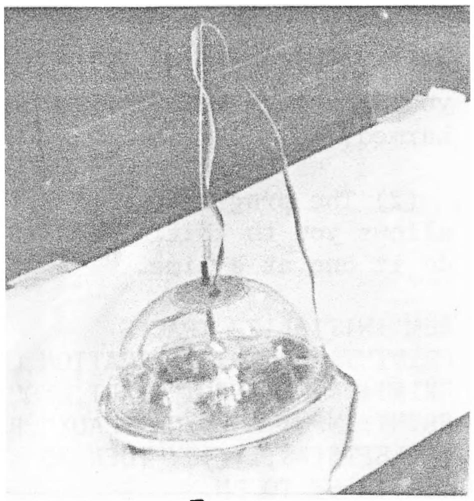
The circuit has only 3 modes of operation: full forward ("GO" signal high), full reverse ("GO" and "DIR" both high), or completely stopped ("GO" low).The transistors will be alternately in cutoff or saturation, thus connecting the motor either way around.

It therefore only takes 4 lines to control 2 motors, with the computer logic determining their direction.Thus the robot can travel forwards, backwards, or rotate (motors in opposite directions.)

FIG. 2







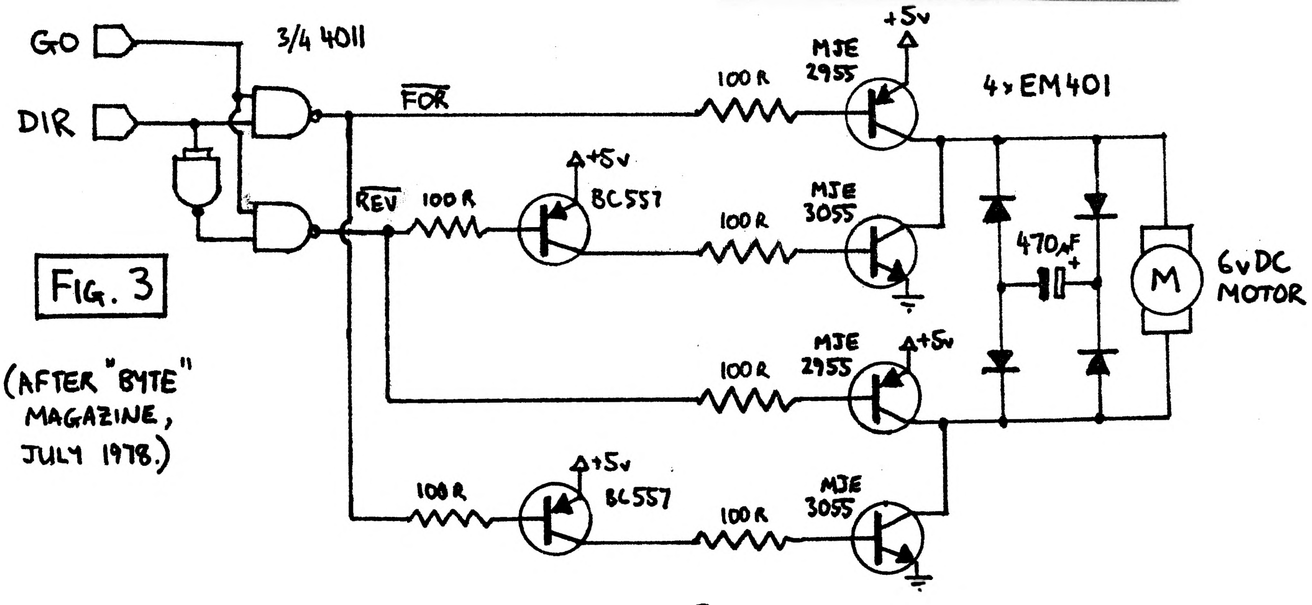


FIG. 3

(AFTER "BYTE" MAGAZINE, JULY 1978.)

COMPONENTS AND AVAILABILITY:

Two 6v DC hobby motors with gearboxes (from toy or hobby shops, eg. "The Model Dockyard", Swanston St. Expensive, but strong and reliable and plenty of torque.)
Wheels and drive chains or belts to suit. (A slippable drive belt is a useful safety clutch for these motors.)
Circular aluminium chassis, approx. 400mm diameter.
Clear plastic dome cover (eg. K mart fish bowl / terrarium $4.00)
6v solenoid, to actuate the pen (eg. an old relay)
Speaker (eg. from a discarded transistor radio)
Three microswitches, for impact ring
Aluminium strip, bent into a ring slightly larger than the chassis, mounted on springy supports.
Torch globes, LEDs etc.

****

If you have taken the plunge and up graded to a 5.25 inch floppy disk, then you are probably aware of the shortcomings of the standard OSI operating system. There is a new DOS coming soon, but in the meantime, here are a couple of ideas which could make life a little easier.

(1) Have you ever written a program and then discovered that, either the created file does not have enough tracks, or you have forgotten to create a file for it. You could do a PUT by track number but if you wanted to create a file name in the directory later, the create utility would write nulls all over your program. To avoid this, there is a simple remedy:
add this line to the create program and resave it:-

```
20265 INPUT'INITIALIZE TRACKS';IN$:IF IN$  'Y'THEN20320
```

If you do not respond to the question with 'Y' then your program will not be harmed.

(2) The program listed below could be added to your utilities disk, it allows you to INitialize any number of tracks on a disk, without having to do it one at a time.

```
10 REM INITIALIZE TRACKS
20 PRINT"TRACK INITIALIZATION UTILITY"
30 PRINT:PRINT:INPUT"START,END";ST,EN
32 PRINT:INPUT"ARE YOU READY";R$
35 IF LEFT$(R$,1)<>"Y"THEN 30
40 FOR IN=ST TO EN
50 IN$=RIGHT$(STR$(I+),2)
52 IF IN<10 THEN IN$="0"+RIGHT$(IN$,1)
60 DISK!"IN"+IN$
70 NEXT IN:END
```

Steve Stokes

****

JOYSTICKS:

If there are any people wanting joysticks, KAOS members can now buy single ATARI joysticks for $14 each. If you are interested then contact Kelvin Eldridge on ... ----.

DEMO PROGRAM
(see article on page 12)

```
 5 VB=0:REM C1-P
 10 IFPEEK(57088)<128THENVB=1:REM C4-P
 15 WI=PEEK(65505) :REM CHARACTERS ACROSS SCREEN
 20 LL=32:REM C1-P LINE LENGTH
 25 IFWI>32THENLL=64
 30 C1=54016+PEEK(65504):REM BOTTOM LEFT CORNER
 35 IFPEEK(65506)THENC1=55040+PEEK(65504)
 40 REM TEST FOR DISK
 45 IFPEEK(11828)=66ANDPEEK(11829)=65THENDI=1:REM DISK FLAG
 47 IFPEEK(542)=153ANDPEEK(543)=37THENDI=0:REM PICO DOS SAME AS CASSETTE
 50 GOSUB1000:A$="DISK AND ROM DEMO":GOSUB6000:PRINT:PRINT:PRINT:PRINT
 55 PRINT"DO YOU WANT INSTRUCTIONS":GOSUB2000:IFA=89THENGOSUB5000
 60 REM TURN OFF CONTROL C
 65 IFDITHENPOKE2073,96:GOTO100
 70 POKE530,1
 100 GOSUB1000:GOSUB3000:X=1:Y=23
 110 A=154:GOSUB500:GOSUB2000:IFA=13THENY=Y-1:X=1:GOTO110
 120 IFA=127THENGOTO100
 130 IFA=10THEN110
 140 IFA=8THENX=1:Y=23:GOTO110
 150 IFA=27THEN170
 160 GOSUB500:X=X+1:GOTO110
 170 GOSUB1000:IFDITHENPOKE2073,173:RUN"BEXEC*
 180 POKE530,0:END
 500 IFX=WITHENY=Y-1:X=1
 510 IFY=0THENY=23
 520 POKEC1-(LL*Y)+X,A:RETURN
 999 END
 1000 REM CLEAR SCREEN  FOR DABUG MONITORS
 1005 IFDITHENDISK!"GO FCD5":GOTO1015
 1010 PRINTCHR$(127):POKE11,213:POKE12,252:V=USR(V)
 1015 RETURN
 2000 REM GET CHARACTER FROM KEYBOARD
 2005 IFDITHENGOTO2015
 2010 POKE11,00:POKE12,253:X=USR(X):A=PEEK(531):GOTO2020
 2015 DISK!"GO 252B":A=PEEK(9804):IFVBORWI>50THENA=PEEK(9815)
 2020 RETURN
 3000 REM DRAW BORDER
 3005 FORX=0TOWI:POKEC1+X,161:NEXTX
 3010 FORY=1TO23:POKEC1-(LL*Y),161:POKEC1-(LL*Y)+WI,161:NEXTY
 3015 FORX=0TOWI:POKEC1-(LL*Y)+X,161:NEXTX
 3020 RETURN
 5000 GOSUB1000: FORH=0TO18:READA$:GOSUB6000:NEXTH
 5010 GOSUB3000:GOSUB2000:RETURN
 5100 DATA"UP- GRADE DEMO","","THIS PROGRAM ECHOS THE"
 5105 DATA"KEY PRESSED TO THE","SCREEN WITH THE FOLLOWING"
 5110 DATA"EXCEPTIONS","","RUBOUT - CLEARS SCREEN"
 5115 DATA"CTRL H - HOME CURSOR","RETURN - START NEXT LINE
 5120 DATA"ESC - EXIT TO BEXEC*","     OR IMMED MODE"
 5130 DATA" "," "," ","PRESS A KEY TO CONTINUE"
 5140 DATA" "," "," "
 6000 REM CENTRES INSTRUCTIONS IN MIDDLE OF SCREEN
 6005 TB=INT((WI-LEN(A$))/2)
 6010 PRINTTAB(TB)A$:RETURN
OK
```

This month we look at ways of making software compatable with all OHIO systems, Disk or Cassette. To do this we need a simple way of finding which keyboard , screen size , cursor start position your machine uses, and whether you are using cassette or disk, OS65D3 or PICO DOS.

The simplest most commonly used test is to PEEK the keyboard and compare the return value with 127. If the value is greater than 127 then you have a C1-P keyboard. It stands to reason that if you do not have the above keyboard then you have the C4-P keyboard.

The next three tests use locations in the monitor ROM which tell the screen print routine the screen size etc. The first location of interest is 65505 , this location contains the number of characters across the screen - 1 , eg. 65505 contains 23, this is a 24 wide screen. The second location, 65504 , tells you whether there is 1K or 2K of screen RAM eg.Not 0 = 2K of screen RAM and 0 = 1K of screen RAM. This second test result is used by the third test to set the cursor start position. If the screen size is 1K then 54016 is added to the value from 65506 to get the cursor start position, if the screen is not 1K then they add 55040 to the value from 65506 .

Another variable which is used and that has to be defined is the line length . This is a value that is used to get from one line to the next. To define this we compare the number of characters across the screen with 32, if it greater then LL (Line Length) =64, otherwise LL=32 .

To test for cassette or disk you have to find a memory location that is only used for the operation being tested. One small note, the main difference between PICO DOS and Cassette is that PICO DOS modifies the LOAD and SAVE vectors. Also when using disk BASIC it loads off disk and resides from 512 upwards. The easiest method of testing for cassette and PICO DOS by ckecking for a '0' at the start of the BASIC workspace eg PEEK(768).

You should now be able to use a series of PEEKs to set up a game to enable it to work on any OHIO system (with a polled keyboard).

You could also test to see if the machine has a DABUG with a PEEK, for screen clears etc, but I will leave this one for you to find .

To give you some idea how to do this I have included a demo program that has no real purpose other than to show you how it can be done.

Jeff Rae

****

ATARI SOFTWARE REVIEW

QUEST FOR POWER Review by Dan Chaffets - Reprinted from A.C.E. newsletter

This is the most exciting adventure game I have ever played, the game uses fine scrolling to create an image of actually taking part in the adventure. The graphics are very impressive, it has two, three dimensional pictures that are very realistic. The new Crystalsonics are very different to anything I have heard yet, and they include a new reverb which is incredible.

As the adventure began, I was summoned to Camelot and was then told by Arthur that I was supposed to find the scroll of Truth and bring it back to Camelot. "Piece of cake", I thought to myself as I began my journey, little did I know I was in for one hell of a surprise.

All of a sudden I saw this huge fire breathing dragon, I greeted it, but the dragon charged me, this was definitly a hostile reaction on the part of the dragon. Since I didn't want to tangle with five tons of dragon, I decided to run. I ducked in to a near by cave and found that it was extremely dark (very scary), I ran back out again. After losing the dragon and finding enough gold I went to Leeds and bought a boat and a lamp. With the lamp I was able to explore the caves of Somerset and I found some interesting things there.

I continued exploring and came upon the Black Forest. This place was extremely weird, all of a sudden this thing started running towards me. It turned out to be the evil giant Gogmagog, I turned to run, but it was too late, the Gogmagog was already on me. She picked me up (it must be a she to be that nasty) and started choking me. Deciding to wreck her dinner plans, I slapped her across the face with my left hand, she fell backwards and I finished her off with a karate chop to the throat, collapsing her trachea. Now that I had enough magical power to get the magic scroll of Truth, I proceeded to find it.

This is only a small portion of this fantastic program. I left most of it out because it is no fun to play if you know what is going to happen. I can recommend this program to anyone.

****

THE SYM SECTION

A new group within KAOS has been formed to serve the interests of SYM owners.

The first aim is to share knowledge, interests and resourses. For this reason, we would like all SYM owners to fill in a form, which is available from KAOS upon request, to enable us to get an idea of what people want.

We will create a library of software, hardware circuits, books and magazines which will be a part of the general KAOS library ie. a SYM E port to the TASKER BUS interface will appear in the newsletter shortly. We also hope to arrange for discounts with local suppliers.

EPROM PROGRAMMER FOR SYM USERS.

A new 2716 eprom programmer is available from CSS 38 Fawkner St. Essendon 3040. It mainly resides in software and runs off the AA socket. An extra 6522 is required plus a 25 volt power source. When the program is run a program from any part of the SYM can copied into the eprom. Each line is checked after it has been blown and the program stops if an error is detected. The cost is $65.00 and it made up ready to go with full instructions.

FOR SALE

AIM 65 with BASIC and ASSEMBLER, in case with power supply. Hardly used, $900.00 . Ring R. Veerman on [illegible] for more details.

****

Here is yet another patch to fix the DOS on C4-Ps which have a DABUG in ROM. These POKEs can be add to the BEXEC* program or run as separate program, change the address of the serial port to suit the DABUG ie. from $FC00 to $F000.

```
10X=240:POKE9424,X:POKE9432,X:POKE9436,X:POKE9464,X
20POKE9473,X:POKE9816,1
```

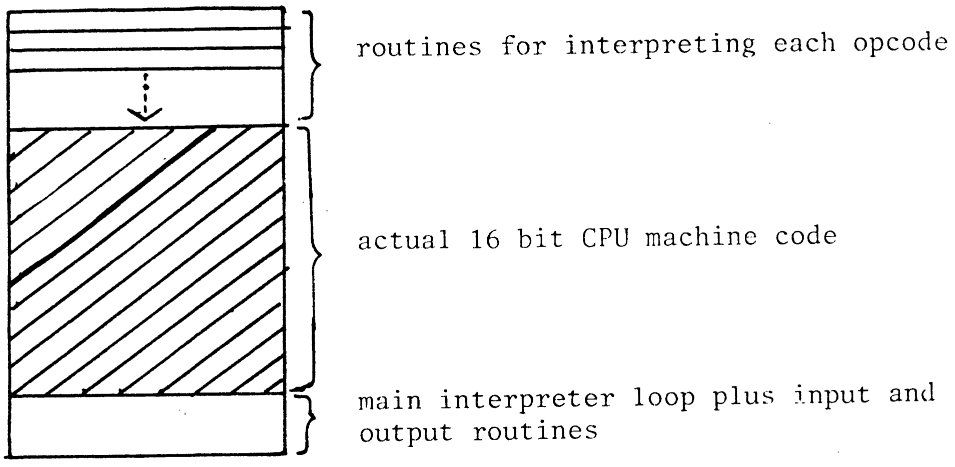
Jeff Rae.

THE 16 BIT PSEUDO MACHINE

Three to four years ago, 16 Bit CPU's were an exception and 8 Bit CPU's were just starting to establish themselves. Most people wished they could change to the more powerful 16 Bit CPU's, but due to cost this was not possible. So in defiance of this limitation a few clever people decided to do something about it. The solution they came up with was that since any microcomputer's or any computers CPU instruction set was only a program written in microcode which turned on appropriate gates. Why couldn't a more powerful instruction set usingthe existing instruction set of 8 Bit micro's be written? No reason at all! So into existence came 16 Bit instruction sets, coded in Z80, 6502, 8080, etc. machines. Some of the ones you might be aware of are P-CODE, SWEET 16, I2L and our yet unnamed 16 Bit machine lining in the heart of our OSI assembler.

The concept of a 16 Bit Pseudo Machine is very close to that of a Basic Interpreter. Like Basic which has pre-defined instructions, such as 'PRINT', 'INPUT', etc. we have machine code instructions such as 'CMC','DADD,etc., which are interpreted one at a time by 'REAL' machine code routine, written in the 'TRUE' machine language. Just like Basic, it provides us with a more powerful software tool, but at the expense of speed.

The structure of the 'Pseudo' interpreter is shown in the illustration.


routines for interpreting each opcode
actual 16 bit CPU machine code
main interpreter loop plus input and output routines

Within the various Pseudo machines there are different philosophies, some are written as a totally new machine code language, one of these is a P-code interpreter. If you haven't heard of it before it is a piece of software which has been written for most micro's, which allows the user to run a very powerful system called UCSD, Pascal and Fortran. Because most of UCSD software is written in P-code all the manufacturer has to do is write a P-code interpreter for his micro, define the input and output handlers and he can then run all the UCSD software that has been written. This is a very advantageous concept which most companies have jumped on.

The other technique used is that instead of replacing the entire instruction set, people have written their Pseudo machines so that they suppliment the existing machine code language. Sweet 16 and the OSI Assembler interpreter are two examples of this type.

For those of you who may not know, SWEET 16 is a small pseudo interpreter written for the Apple. The Apple company have made use of this piece of software through-out their software. eg.

```
          LDA  IN,Y
          CMP  "M"
          BNE  NOMOVE
          JSR  SW16    Jump to Sweet 16 interpreter
MLOOP     LD   @R1     R1 holds source address
          ST   @R2     R2 holds destination address
          DCR  R3      decrement length
          BNZ  MLOOP   loop until done
          RTN          return to normal 6502 code
NOMOVE    CMP  "E"
          BEQ  EXIT
          INY
```

The above 'BLOCK MOVE' piece of code only takes six bytes in SWEET 16 code, whereas if you wrote in 6502 code it would take four to five times more code.

The OSI Assembler interpreter works in a very similar fashion to SWEET 16 but it is a lot more powerful!

Using this method there is no reason why you cannot simulate any computers' instruction set, anything from a Z80, 6809, 68000, etc,etc Of course you must not expect it to be as fast,

Imagine being able to write and run in the code of any machine, or maybe more than one machine, changing the hardware is a lot more expensive than just loading a cassette or disk!

George Nikolaidis

++

ATARI NEWS

Welcome again to Atari news, it only seems a few days since I dragged the last one from my typewriter.

What's new this month? Well, something amazing has happened. After my gripe last month about the non-delivery of our Atari technical manuals, we received them. They were post marked two days after the KAOS newsletter came out. 'Hail' to the power of KAOS.

If anyone wants to convert their Atari 810 disk drive to the fast format I now have the conversion details. You must have access to an Eprom burner, but is well worth the hassle as it makes the Atari work 40% faster. Still on the 810, there is a new data separator which will stop most of those annoying 144 errors. There are two ways of getting it done, Atari will do the conversion for a mere $150.00 or send $30.00, plus postage to THE PERCOM DATA COMPANY, INC. 211 KIRBY ST, GARLAND, TEXAS 75042, U.S.A.

If anybody is interested in FORTH, we have three versions for the Atari. We also have the complete User group library on disk. For a copy send a formatted disk to me and I will send it back with a program called FILEINDX, this list every program we have.

We now have the Atari word processor, it would be the best packaged and presented software I've seen. It's about three inches thick! You get two master W.P. disks and one training disk, also there is a step by step audio tape that takes you through the book. (George has written a review but there is no space for it in this months' newsletter.) There review copy was supplied by Futuretronics. They also loaned us a copy of the soon to be released PILOT cartridge which is designed to teach children about personal computing. We will do a full review of it when it is released. We found it difficult to follow as we only had a copy of the designers' work book.

RUMOURS AND FACTS by Mark Benioff A.C.E.

Atari Inc. has very few software projects for release in 1982, Pascal should be released "finally" early in 1982 Micro Soft Basic will be available. First of all on disk (you will need 32K to run it) then later it will be released on two 8K Roms for the 800.The people with the Atari 400 will miss out.

The Monkey Wrench is a machine language Rom which uses the right hand slot on the Atari 800. It is a utility program which has 9 basic commands and a machine language monitor with 15 commands to interact with the powerful features of the 6502 microprocessor. The 9 basic commands are Auto line numbering, Delete line numbers, Change margins, Memory test renumberer, Cursor exchange, Hex conversion, Decimal conversion and Monitor. The Monitor features are display memoryy, interrogate memory, display registers, alter memory, alter 6502 registers, GOTO, EXIT, Save memory to cassette, Load memory from cassette, Error, Hunt for ASCII string, Hunt for Hex characters, Disassemble listing, Calculate Branch and Atari Dos.

The Monkey Wrench comes on a cartridge with a 4K Eprom and is set up to take one more 4K Eprom. It can be used with or with-out the Basic cartridge, it uses memory from Hex 8000 to Hex 9FFF, so when in the machine it uses 8K of memory.
We have a full set of Atari manuals and schematics, if you would like to read them, ring me on

That's all for now,
Gerry

++

While rewriting programs for the C4-P disk system I became very annoyed with POKE 56832,1 syndrome because most programs change the screen format to 32 * 32.

To overcome this problem I wrote a small patch to the DOS which enable you to change the screen size by using a single key press. To help in remembering what the control code was I decided to use the same code as the series 2 C1-P ie CONTROL B.

Load in the Assembler and type in the mnemonics of the following listing.

Happy swapping?!!!

Jeff Rae

```
10                     ;SCREEN SWAP FOR C4-P (CTRLB)
20  258A                   *=$258A
30  258A 4CBD24            JMP PATCH
40  258D EA                NOP
50  24BD                   *=$24BD
60  24BD C910     PATCH    CMP #$10
70  24BF D003              BNE CTRLB
80  24C1 4C8E25            JMP $258E
90  24C4 C902     CTRLB    CMP #$02
100 24C6 D010              BNE EXIT
110 24C8 A000              LDY #$00
120 24CA ADD924            LDA FLAG
130 24CD D001              BNE STORE
140 24CF C8                INY
150 24D0 8C00DE   STORE    STY $DE00
160 24D3 8CD924            STY FLAG
170 24D6 A900              LDA #$00
180 24D8 60       EXIT     RTS
190 24D9 01       FLAG      BYTE 01
```

```
++++++++++++++++++++++++++++++
+                            +
+         FOR SALE           +
+       MICROLINE 80         +
+         $450.00            +
+                            +
+         contact            +
+     Gerry McCaughey        +
+       ph.                  +
+                            +
++++++++++++++++++++++++++++++
```